HDC

Muy buen día a todos!

Hoy me gustaría contaros una historia que me ocurrió hace un tiempo (que podría ser o no real). Esta historia comienza cuando un Jueves, mientras navegaba como de costumbre por Facebook un nuevo alumno de ***HDC*** llamado ***Antonio*** me abre un chat privado y me dice lo siguiente.

Entonces me descargo el archivo, pero con un poco de desconfianza y…

“Espera, no nos interesan tus historias, ¡solo queremos aprender nuevas cosas!”

Tranquilo Manolo, no te impacientes, te adelanto que el archivo que me mandó es un backdoor hecho con NovaLite, así que puedes crear un backdoor y seguirnos el paso, puedes estar seguro que hoy también vas a aprender.

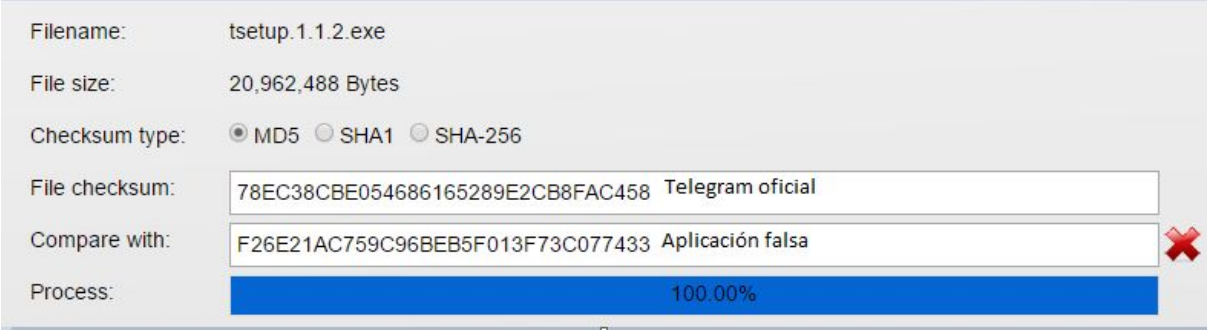
Sigamos con la historia, como ya os he dicho sospeché del ejecutable, por eso me decidí a analizarlo. Las técnicas que vamos a utilizar son parte de una rama de la informática a la que llamamos **informática forense**, esta técnica consiste en analizar los diferentes tipos de datos que hay en una máquina, y la manipulación de estos, también incluye otras técnicas como la recuperación de datos perdidos que veremos en un futuro.

Es aconsejable hacer uso de una máquina virtual para este tipo de análisis.

Ya tenemos nuestro ejecutable descargado, y como hemos sospechado, está en nuestra máquina virtual con Windows.

Vamos a empezar analizando a través de técnicas que no necesiten abrir el archivo, lo primero que vamos a hacer es comparar el hash, si recordáis los hashes, además de para "esconder" las contraseñas, servían para verificar la veracidad de un archivo. Normalmente las aplicaciones oficiales al descargarlas te muestra en la web un hash **MD5** o **SHA-1**, en este caso no es así, pero descargaremos la aplicación y compararemos los hashes.

Podemos obtener el hash **MD5** de un ejecutable en esta web: http://onlinemd5.com/



Vemos que los hashes no coinciden, por lo tanto, no es la aplicación que nuestro amigo nos había prometido.

"Vale, pero eso no quiere decir que sea un malware, podría ser cualquier cosa."

Bueno Manolo, no quiere decir que sea un malware, pero tiene muchas probabilidades, si descubres que la aplicación no es lo que debería ser, quiere decir que nuestro amigo está usando ingeniería social para hacernos abrir algo que en otro caso no abriríamos. En todo caso, vamos a seguir hacia delante.

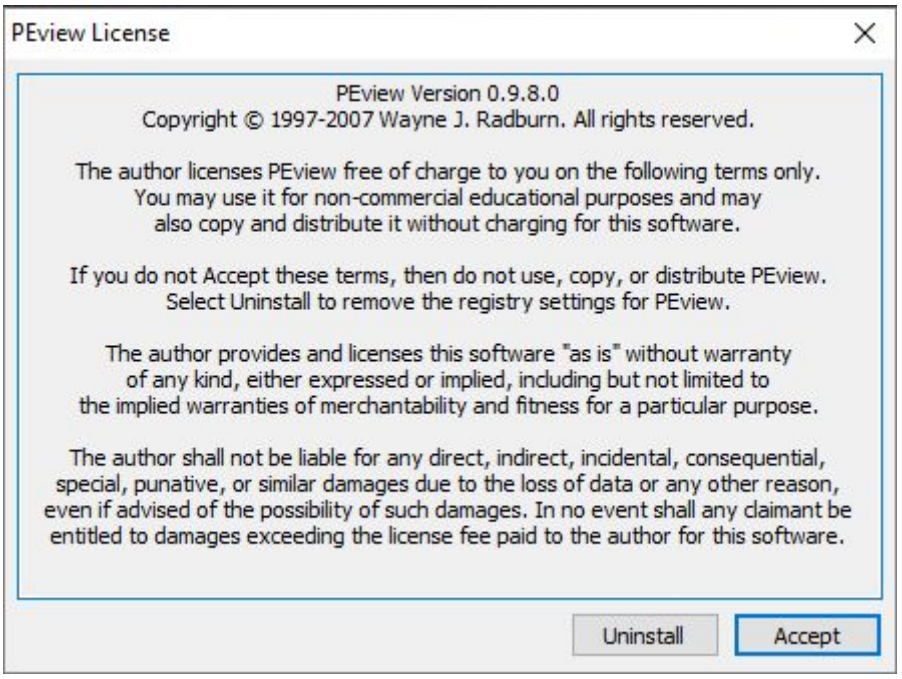
Algo que nos ayudará a saber si un archivo es un malware o no es analizar el archivo con un antivirus, hay páginas webs que se encargan de analizarlo con muchos antivirus diferentes, nosotros lo haremos con esta: https://www.virustotal.com/es/

Como vemos *42 de 59* antivirus lo han detectado, esto creo que acrecienta nuestras sospechas.
Como dato extra os comento que VT (VirusTotal) vende las muestras que se suben de malware a otras empresas encargadas de hacer antivirus para que estas lo analicen, por lo tanto no es aconsejable subir un malware que hayamos hecho nosotros para ver si es indetectable, en su lugar podemos usar otras páginas como http://stopvir.us/ o http://pscan.xyz/.

Sigamos analizando nuestro archivo, hay otra cosa muy interesante que podemos hacer antes de abrirlo para ver que hace. Vamos a ver qué **DLLs** usa y a que funciones llama, para esto vamos a usar **PEview**, que podemos descargarlo de aquí: http://wjradburn.com/software/

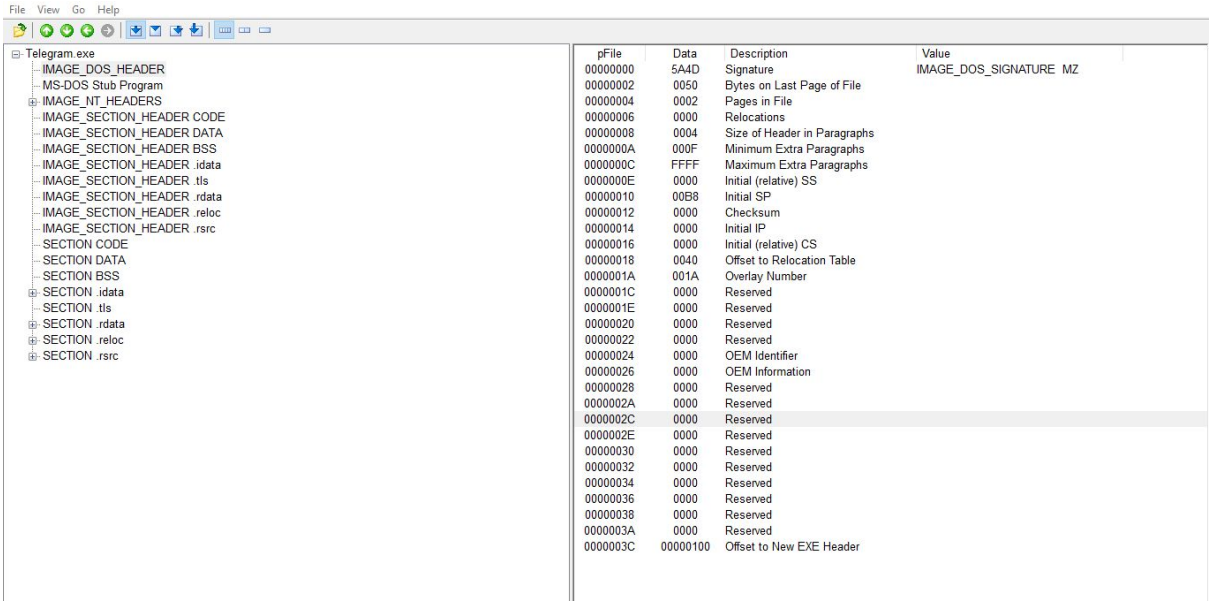
PEview es un software que se usa para analizar **archivos PE**, que son ejecutable portables, del inglés (**Portable Executable**).

Una vez descargado lo abrimos y aceptamos la licencia.

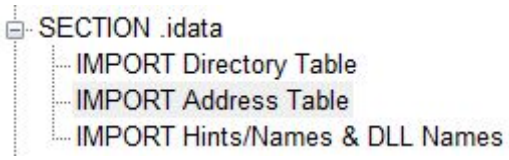


La primera vez que lo abrimos nos pedirá que seleccionemos un archivo, para las próximas veces podemos elegir el archivo en la pestaña: *file >> open.*

Una vez abierto el programa nos encontraremos algo como esto.



Podemos ver que a la izquierda tenemos las diferentes secciones, y a la derecha la información respectiva. De momento podemos ver que hay muchas secciones diferentes, pero a nosotros nos interesa la sección **.idata**, aquí es donde los archivos ejecutables guardan las llamadas al sistema en Windows, vamos a abrirlo y veamos que hay.



Veamos que es cada una:

IMPORT Directory Table: Es la tabla de directorios, para que nos entendamos es una tabla que muestra todos los directorios a los que llamará nuestro ejecutable. Veamos qué contenido tiene.

pFile	Data	Description	Value
00005A00	00000000	Import Name Table RVA	
00005A04	00000000	Time Date Stamp	
00005A08	00000000	Forwarder Chain	
00005A0C	0000920C	Name RVA	kernel32.dll
00005A10	000090DC	Import Address Table RVA	
00005A14	00000000	Import Name Table RVA	
00005A18	00000000	Time Date Stamp	
00005A1C	00000000	Forwarder Chain	
00005A20	000093AA	Name RVA	user32.dll
00005A24	00009138	Import Address Table RVA	
00005A28	00000000	Import Name Table RVA	
00005A2C	00000000	Time Date Stamp	
00005A30	00000000	Forwarder Chain	
00005A34	000093E2	Name RVA	advapi32.dll
00005A38	00009148	Import Address Table RVA	
00005A3C	00000000	Import Name Table RVA	
00005A40	00000000	Time Date Stamp	
00005A44	00000000	Forwarder Chain	
00005A48	00009422	Name RVA	oleaut32.dll
00005A4C	00009158	Import Address Table RVA	
00005A50	00000000	Import Name Table RVA	
00005A54	00000000	Time Date Stamp	
00005A58	00000000	Forwarder Chain	
00005A5C	0000946A	Name RVA	kernel32.dll
00005A60	00009168	Import Address Table RVA	
00005A64	00000000	Import Name Table RVA	
00005A68	00000000	Time Date Stamp	
00005A6C	00000000	Forwarder Chain	
00005A70	000094B6	Name RVA	advapi32.dll
00005A74	0000917C	Import Address Table RVA	
00005A78	00000000	Import Name Table RVA	
00005A7C	00000000	Time Date Stamp	
00005A80	00000000	Forwarder Chain	
00005A84	000094F4	Name RVA	kernel32.dll

“Pero yo no se que hace cada .dll, además hay varios que se han llamado varias veces.”

Bueno ahora veremos con más detalle para que sirve cada dll, y el hecho de que haya algunos que se llaman varias veces es debido a que el programa usa más de una función de ese **dll**. De momento vayamos a la siguiente sección.

IMPORT Address Table: Esta es la tabla de direcciones, muestra cada una de las funciones a las que llama el programa, y las ordena respectivamente con cada **dll**. Veamos que nos encontramos aquí.

pFile	Data	Description	Value
00005B34	00000000	End of Imports	kernel32.dll
00005B38	000093B6	Hint/Name RVA	0000 GetKeyboardType
00005B3C	000093C8	Hint/Name RVA	0000 MessageBoxA
00005B40	000093D6	Hint/Name RVA	0000 CharNextA
00005B44	00000000	End of Imports	user32.dll
00005B48	000093F0	Hint/Name RVA	0000 RegQueryValueExA
00005B4C	00009404	Hint/Name RVA	0000 RegOpenKeyExA
00005B50	00009414	Hint/Name RVA	0000 RegCloseKey
00005B54	00000000	End of Imports	advapi32.dll
00005B58	00009430	Hint/Name RVA	0000 SysFreeString
00005B5C	00009440	Hint/Name RVA	0000 SysReAllocStringLen
00005B60	00009456	Hint/Name RVA	0000 SysAllocStringLen
00005B64	00000000	End of Imports	oleaut32.dll
00005B68	00009478	Hint/Name RVA	0000 TlsSetValue
00005B6C	00009486	Hint/Name RVA	0000 TlsGetValue
00005B70	00009494	Hint/Name RVA	0000 LocalAlloc
00005B74	000094A2	Hint/Name RVA	0000 GetModuleHandleA
00005B78	00000000	End of Imports	kernel32.dll
00005B7C	000094C4	Hint/Name RVA	0000 RegSetValueExA
00005B80	000094D6	Hint/Name RVA	0000 RegCreateKeyA
00005B84	000094E6	Hint/Name RVA	0000 RegCloseKey
00005B88	00000000	End of Imports	advapi32.dll
00005B8C	00009502	Hint/Name RVA	0000 lstrcmpiA
00005B90	0000950E	Hint/Name RVA	0000 WriteProcessMemory
00005B94	00009524	Hint/Name RVA	0000 WaitForSingleObject
00005B98	0000953A	Hint/Name RVA	0000 VirtualProtect
00005B9C	0000954C	Hint/Name RVA	0000 VirtualFree
00005BA0	0000955A	Hint/Name RVA	0000 VirtualAllocEx
00005BA4	0000956C	Hint/Name RVA	0000 VirtualAlloc
00005BA8	0000957C	Hint/Name RVA	0000 Sleep
00005BAC	00009584	Hint/Name RVA	0000 SizeofResource
00005BB0	00009596	Hint/Name RVA	0000 ReadProcessMemory
00005BB4	000095AA	Hint/Name RVA	0000 LockResource

Veamos algunas de ellas y comentemoslas:

kernell32.dll

- *ExitProcess*: Se utiliza para finalizar un proceso.
- *WriteFile*: Se utiliza para escribir un fichero.
- *DeleteFileA:* Elimina un fichero.
- *CopyFileA*: Copia un fichero.
- *CreateProcessA*: Inicia un proceso.

advapi32.dll

- *RegCreateKeyA*: Crea una llave en el registro.
- *RegSetValueExA*: Da valor y tipo a una llave del registro.

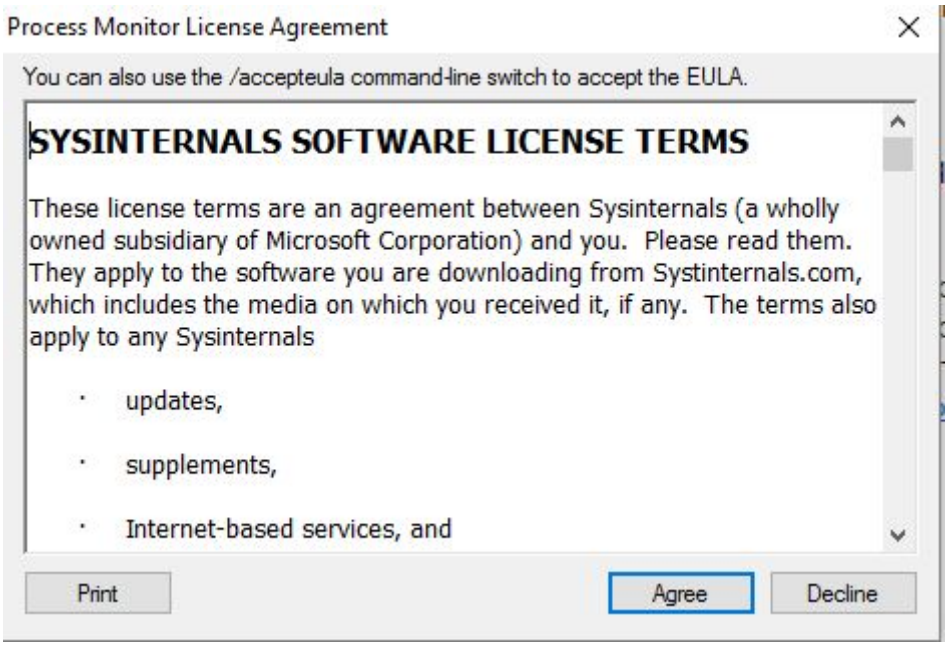
Si se observa bien, estos procesos que hemos comentado son algunos de los que usamos nosotros cuando estábamos creando nuestro backdoor. El ejecutable llama a muchas más funciones que no hemos visto, sería aconsejable ver como funcionan, si se buscan los nombres en Google se encontrará la documentación oficial de Windows que explica cada una de estas funciones (está en inglés).

IMPORT Hints/Names & DLL Names: Muestra los nombres de las funciones y los **DLLs** que ya habíamos visto.

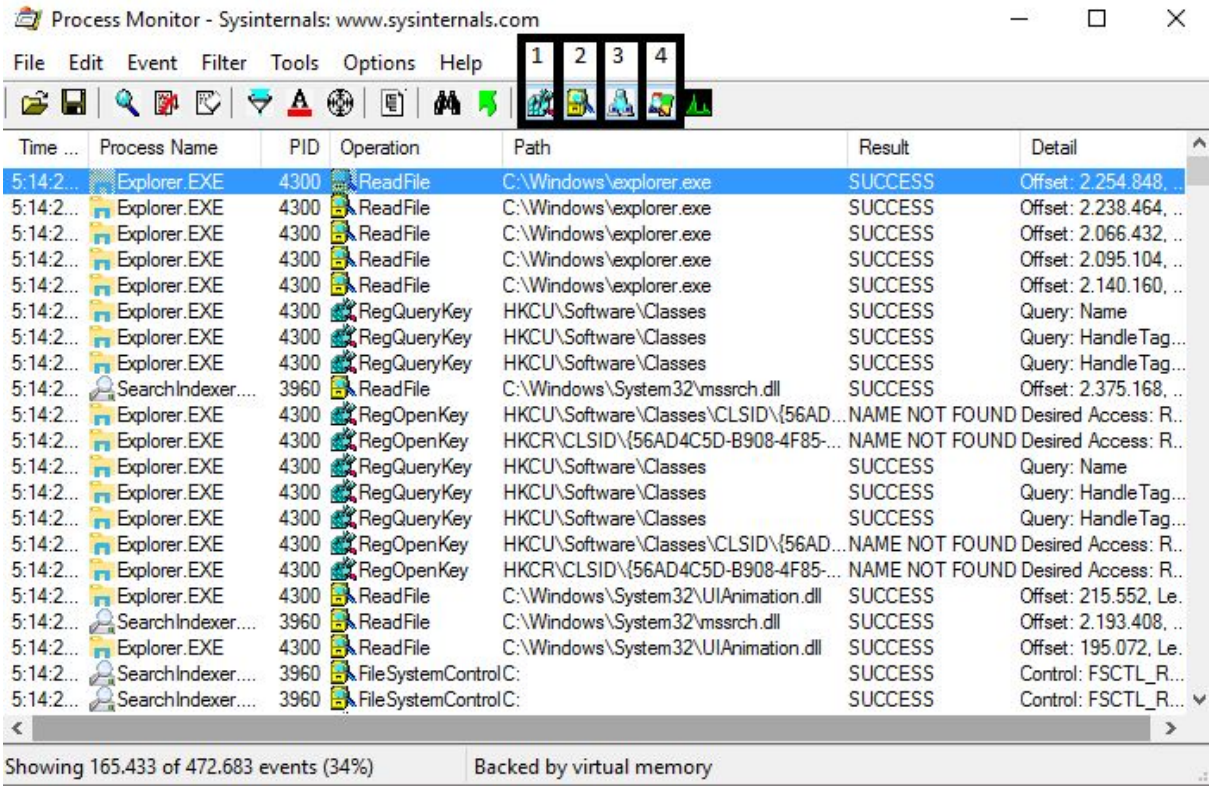
Para continuar con nuestro análisis vamos a tener que abrir nuestro ejecutable, por lo que si no habéis iniciado ya una máquina virtual, es el momento ideal.

Lo que vamos a hacer es ver los cambios que se producen en el registro cuando abrimos el ejecutable, para esto vamos a usar **ProcessMonitor**, que es una aplicación que entre otras funciones permite monitorear la actividad de procesos y del registro de Windows, podemos descargarlo desde aquí:
https://technet.microsoft.com/en-us/sysinternals/processmonitor.aspx

Al descargar y abrir la aplicación vamos a aceptar el acuerdo de licencia.



Y nos encontraremos algo como esto.



Veamos que muestra cada una de las pestañas que hay:

- **Time**: El momento en el que se hizo la operación.

- **Process Name**: El nombre de la operación que ha realizado la operación.

- **PID**: El PID del proceso que ha realizado la operación.

- **Operation**: Operación realizada, las casillas que he marcado sirven cada una para monitorizar un tipo de proceso diferente.

    1. Monitoriza cambios en el registro.
    2. Monitoriza acciones en los ficheros.
    3. Monitoriza la actividad de red. (Esto también se puede hacer con el comando netstat que dimos en la clase 56, pueden probar)
    4. Monitoria procesos y threads.

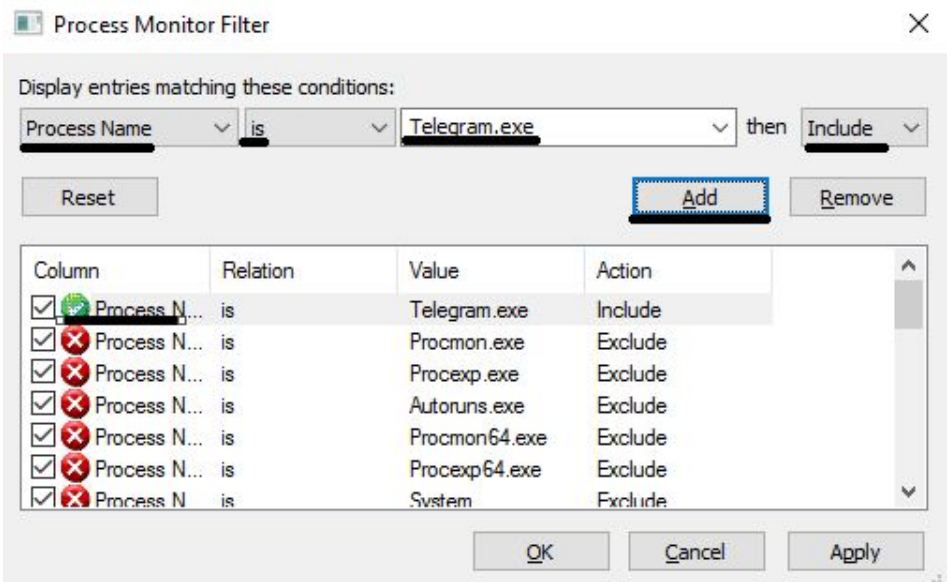
- **Path**: Ruta de la operación.

- **Result**: Resultado.

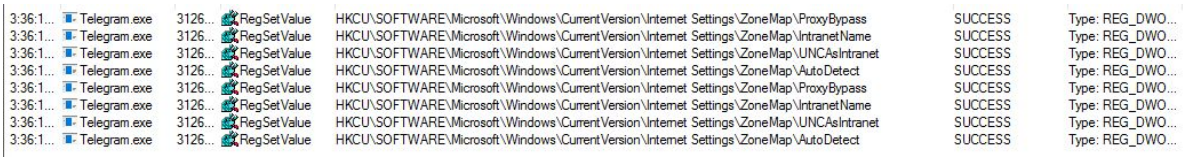
- **Detail**: Detalles extras.

Además podemos añadir filtros, para que nos resulte más fácil encontrar lo que buscamos, para eso tenemos que pulsar las teclas **Ctr+L**. Vamos a empezar viendo qué hace nuestro programa con el registro, para eso vamos a crear un filtro y vamos a seleccionar la pestaña correspondiente.

El filtro que vamos a elegir debe ser **Process Name** y lo vamos a igualar al nombre de nuestro ejecutable. Además crearemos otro filtro, haciendo que action sea igual a **RegSetValue**, que si recordamos de arriba era la forma de añadir un valor al registro.

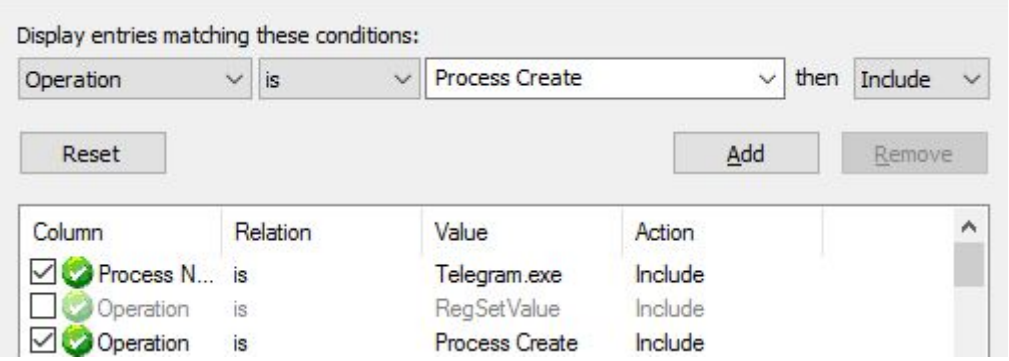


Ya podemos iniciar el programa para ver los cambios que hace en el registro.



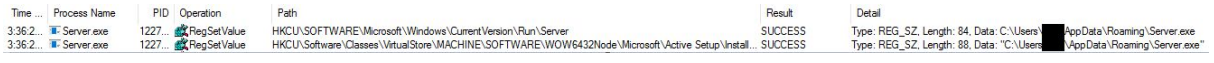
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS	Type: REG_DWO...
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName	SUCCESS	Type: REG_DWO...
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet	SUCCESS	Type: REG_DWO...
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect	SUCCESS	Type: REG_DWO...
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS	Type: REG_DWO...
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName	SUCCESS	Type: REG_DWO...
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet	SUCCESS	Type: REG_DWO...
3:36:1...	Telegram.exe	3126...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect	SUCCESS	Type: REG_DWO...

Vemos que solo se han añadido claves al registro en esta ruta *“HKCU/Software/Microsoft/Windows/CurrentVersion/InternetSettings/ZoneMap”* Estas claves modificadas del registro permiten a un ejecutable conocer los datos de la zona a pesar de que se usen medidas de protección. A pesar de esto, parece que no se ha modificado nada más, quitemos el filtro de **RegSetValue** y monitoricemos los procesos y threads de **Telegram.exe**, vamos a añadir también un filtro que se llame “**Process Create**”, veamos cómo quedan nuestros filtros y los resultados.



Time ...	Process Name	PID	Operation	Path	Result	Detail
3:36:2...	Telegram.exe	3126...	Process Create	C:\Users\[redacted]AppData\Roaming\Server.exe	SUCCESS	PID: 122736, Command line:

Esto parece interesante , añade al inicio de Windows una aplicación llamada **Server.exe**, parece que cada paso que damos nos da una pista más de que lo que nos pasó nuestro amigo es algo malicioso, volvamos a revisar las modificaciones en el registro, pero en este caso de **Server.exe**.



Time ...	Process Name	PID	Operation	Path	Result	Detail
3:36:2...	Server.exe	1227...	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Server	SUCCESS	Type: REG_SZ, Length: 84, Data: C:\Users\[redacted]AppData\Roaming\Server.exe
3:36:2...	Server.exe	1227...	RegSetValue	HKCU\Software\Classes\VirtualStore\MACHINE\SOFTWARE\WOW6432Node\Microsoft\Active Setup\Install...	SUCCESS	Type: REG_SZ, Length: 88, Data: "C:\Users\[redacted]\AppData\Roaming\Server.exe"

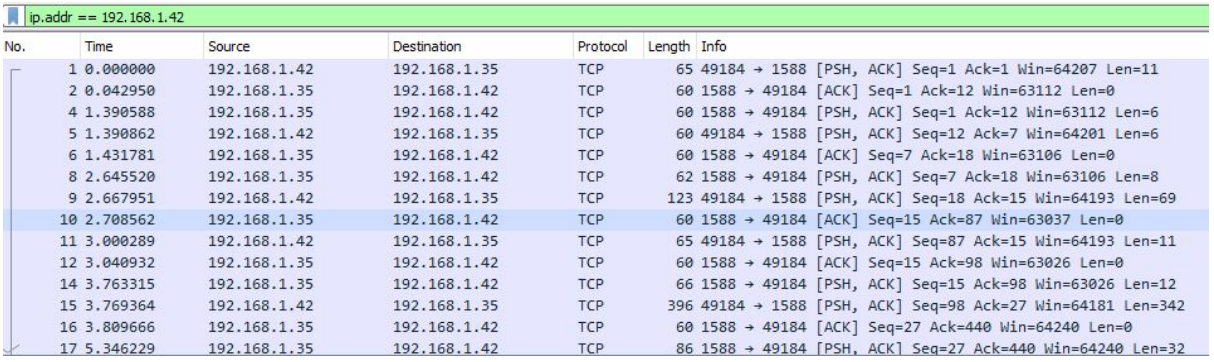
Ahora si podemos ver cómo este ejecutable se añade al registro para iniciarse a sí mismo, en este caso podríamos analizarlo con **PEview**, pero vamos a seguir hacia delante, si ahora intentan observar la actividad de red verán que no hay nada, por lo tanto volvamos a ver qué procesos crea **Server.exe**.

Time ...	Process Name	PID	Operation	Path	Result	Detail
3:36:2...	Server.exe	1227...	Process Create	C:\WINDOWS\SysWOW64\svchost.exe	SUCCESS	PID: 10416, Command line: "svchost.exe" "C:\Users\roll_\AppData\Roaming\Serv...
3:52:0...	Server.exe	3016...	Process Create	C:\WINDOWS\SysWOW64\svchost.exe	SUCCESS	PID: 310200, Command line: "svchost.exe" "C:\Users\roll_\AppData\Roaming\Ser...
4:18:5...	Server.exe	1620...	Process Create	C:\WINDOWS\SysWOW64\svchost.exe	SUCCESS	PID: 240208, Command line: "svchost.exe" "C:\Users\roll_\AppData\Roaming\Ser...
4:24:0...	Server.exe	3305...	Process Create	C:\WINDOWS\SysWOW64\svchost.exe	SUCCESS	PID: 330872, Command line: "svchost.exe" "C:\Users\roll_\AppData\Roaming\Ser...
4:25:1...	Server.exe	3687...	Process Create	C:\WINDOWS\SysWOW64\svchost.exe	SUCCESS	PID: 371448, Command line: "svchost.exe" "C:\Users\roll_\AppData\Roaming\Ser...

Podemos observar que crea un tercer proceso "***svchost.exe***", éste es un proceso del sistema de Windows, por lo que el ejecutable está intentando suplantar su identidad como método para ocultarse, veamos ahora la actividad de red que tiene el proceso svchost.exe, lo haremos creando dos filtros de "*operation*" para ver las conexiones, **TCP Connect y UDP Connect**.

Time ...	Process Name	PID	Operation	Path	Result	Detail
3:36:0...	svchost.exe	1028	TCP Connect	[redacted]	SUCCESS	Length: 0, mss: 1452, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 16616, rcvwinscale: 8, ...
3:36:2...	svchost.exe	10416	TCP Connect		SUCCESS	Length: 0, mss: 1460, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 525600, rcvwinscale: 8...
3:40:4...	svchost.exe	2284	TCP Connect		SUCCESS	Length: 0, mss: 1440, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 16616, rcvwinscale: 8, ...
3:48:2...	svchost.exe	464	TCP Connect		SUCCESS	Length: 0, mss: 1452, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 16616, rcvwinscale: 8, ...
3:52:0...	svchost.exe	3102...	TCP Connect		SUCCESS	Length: 0, mss: 1460, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 525600, rcvwinscale: 8...
3:55:1...	svchost.exe	2284	TCP Connect		SUCCESS	Length: 0, mss: 1440, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 16616, rcvwinscale: 8, ...
3:55:4...	svchost.exe	2284	TCP Connect		SUCCESS	Length: 0, mss: 1440, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 16616, rcvwinscale: 8, ...
3:55:4...	svchost.exe	2284	TCP Connect		SUCCESS	Length: 0, mss: 1440, sackopt: 1, tsopt: 0, wsopt: 1, rcvwin: 16616, rcvwinscale: 8, ...

Esta información también se podría obtener con un comando en consola, sería aconsejable intentar obtenerlo para ver si recordáis bien los comandos. Ahora que tenemos la IP vamos a abrir nuestro Wireshark y vamos a ver cómo se comunican. Para ello vamos a usar el filtro "*ip.addr == x.x.x.x*".



ip.addr == 192.168.1.42

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.42	192.168.1.35	TCP	65	49184 → 1588 [PSH, ACK] Seq=1 Ack=1 Win=64207 Len=11
2	0.042950	192.168.1.35	192.168.1.42	TCP	60	1588 → 49184 [ACK] Seq=1 Ack=12 Win=63112 Len=0
4	1.390588	192.168.1.35	192.168.1.42	TCP	60	1588 → 49184 [PSH, ACK] Seq=1 Ack=12 Win=63112 Len=6
5	1.390862	192.168.1.42	192.168.1.35	TCP	60	49184 → 1588 [PSH, ACK] Seq=12 Ack=7 Win=64201 Len=6
6	1.431781	192.168.1.35	192.168.1.42	TCP	60	1588 → 49184 [ACK] Seq=7 Ack=18 Win=63106 Len=0
8	2.645520	192.168.1.35	192.168.1.42	TCP	62	1588 → 49184 [PSH, ACK] Seq=7 Ack=18 Win=63106 Len=8
9	2.667951	192.168.1.42	192.168.1.35	TCP	123	49184 → 1588 [PSH, ACK] Seq=18 Ack=15 Win=64193 Len=69
10	2.708562	192.168.1.35	192.168.1.42	TCP	60	1588 → 49184 [ACK] Seq=15 Ack=87 Win=63037 Len=0
11	3.000289	192.168.1.42	192.168.1.35	TCP	65	49184 → 1588 [PSH, ACK] Seq=87 Ack=15 Win=64193 Len=11
12	3.040932	192.168.1.35	192.168.1.42	TCP	60	1588 → 49184 [ACK] Seq=15 Ack=98 Win=63026 Len=0
14	3.763315	192.168.1.35	192.168.1.42	TCP	66	1588 → 49184 [PSH, ACK] Seq=15 Ack=98 Win=63026 Len=12
15	3.769364	192.168.1.42	192.168.1.35	TCP	396	49184 → 1588 [PSH, ACK] Seq=98 Ack=27 Win=64181 Len=342
16	3.809666	192.168.1.35	192.168.1.42	TCP	60	1588 → 49184 [ACK] Seq=27 Ack=440 Win=64240 Len=0
17	5.346229	192.168.1.35	192.168.1.42	TCP	86	1588 → 49184 [PSH, ACK] Seq=27 Ack=440 Win=64240 Len=32

El escaneo consiste en una gran cantidad de paquetes **TCP**, suponemos que estos son los que envían información de la víctima al atacante, si nos fijamos bien hay una clase de paquete que se envía del cliente al servidor que incluye la palabra "*CONNECTED?*".

```
> Frame 1: 65 bytes on wire (520 bits), 65 bytes captured (520 bits) on interface 0
> Ethernet II, Src: CadmusCo_fa:d8:3b (08:00:27:fa:d8:3b), Dst: Pegatron_0d:e3:ae (70:54:d2:0d:e3:ae)
> Internet Protocol Version 4, Src: 192.168.1.42, Dst: 192.168.1.35
> Transmission Control Protocol, Src Port: 49184 (49184), Dst Port: 1588 (1588), Seq: 1, Ack: 1, Len: 11
> Data (11 bytes)
```

```
0000  70 54 d2 0d e3 ae 08 00  27 fa d8 3b 08 00 45 00   pT...... '..;..E.
0010  00 33 01 fc 40 00 80 06  00 00 c0 a8 01 2a c0 a8   .3..@... .....*..
0020  01 23 c0 20 06 34 70 14  b5 fa 35 3d 83 0f 50 18   .#. .4p. ..5=..P.
0030  fa cf 83 c3 00 00 43 4f  4e 4e 45 43 54 45 44 3f   ......CO NNECTED?
0040  0a                                                 .
```

Veamos esta cadena de paquetes >> "*Click derecho - Follow - TCP Stream*".

```
CONNECTED?
PING
PONG
drives
drives|C:\| |NTFS|3|D:\| ||5|E:\|VBOX_Desktop|VBoxSharedFolderFS|4|
CONNECTED?
lfiles|C:\
lfiles|Directory||$Recycle.Bin|Directory||Archivos de programa|Directory||Documents and Settings|Directory||PerfLogs|
Directory||Program Files|Directory||Program Files (x86)|Directory||ProgramData|Directory||Recovery|Directory||System Volume
Information|Directory||Users|Directory||Windows|Hidden/System File/Archive|1023.10 MB|pagefile.sys|
lfiles|C:\Program Files (x86)\
lfiles|Directory||.|Directory||..|Directory||Common Files|Directory||Internet Explorer|Directory||MSBuild|Directory||Nueva
carpeta|Directory||Reference Assemblies|Directory||Uninstall Information|Directory||Windows Defender|Directory||Windows Mail|
Directory||Windows Media Player|Directory||Windows NT|Directory||Windows Photo Viewer|Directory||Windows Portable Devices|
Directory||Windows Sidebar|Directory||WinPcap|Hidden/System File/Archive|0.17 KB|desktop.ini|
SHELL|DESACTIVAR
```

Después de preguntar si está conectado se envían dos paquetes "*PING - PONG*", para comprobar que siguen conectados y a partir de ahí se enviarían los comandos:

- **drives**: Pide que se listen las unidades de memoria.
- **lfiles**: Pide que se listen los ficheros.
- **SHELL|DESACTIVAR**: Finaliza la serie de comandos.

Ahora que sabemos esto podemos crear un nuevo filtro en Wireshark para ver cada vez que el atacante se conecta a nosotros, este sería "*ip.addr == x.x.x.x && tcp contains "CONNECTED?"*".

ip.addr == 192.168.1.42 && tcp contains "CONNECTED?"

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.42	192.168.1.35	TCP	65	49184 → 1588 [PSH, ACK] Seq=1 Ack=1 Win=64207 Len=11
11	3.000289	192.168.1.42	192.168.1.35	TCP	65	49184 → 1588 [PSH, ACK] Seq=87 Ack=15 Win=64193 Len=11
37	9.000028	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=1057 Ack=28 Win=64213 Len=11
43	12.000101	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=1074 Ack=34 Win=64207 Len=11
54	14.999974	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=1565 Ack=62 Win=64179 Len=11
61	18.001109	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=2139 Ack=88 Win=64153 Len=11
64	21.000336	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=2150 Ack=88 Win=64153 Len=11
74	24.000070	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=2167 Ack=94 Win=64147 Len=11
79	27.000150	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=2520 Ack=106 Win=64135 Len=11
88	30.001083	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=3055 Ack=184 Win=64057 Len=11
95	33.001213	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=3072 Ack=190 Win=64051 Len=11
97	36.000123	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=3083 Ack=190 Win=64051 Len=11
114	39.001924	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=3094 Ack=244 Win=63997 Len=11
123	42.014211	192.168.1.42	192.168.1.35	TCP	65	49185 → 1588 [PSH, ACK] Seq=3111 Ack=250 Win=63991 Len=11

Por ahora vamos a acabar de analizar el malware por aquí, pero vamos a hacer una recopilación de los datos que hemos obtenido:

1º Hash diferente de la aplicación oficial.

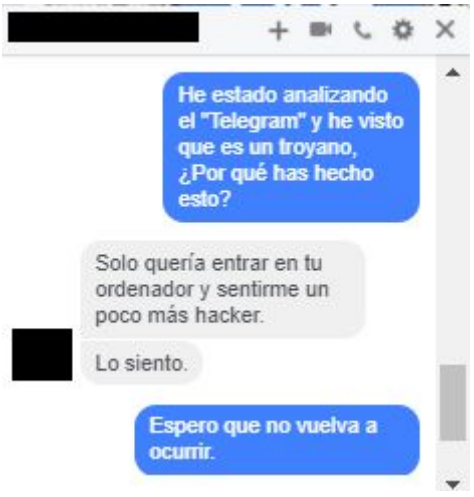
2º 42/59 antivirus lo detectan como malware.

3º Puede modificar procesos, archivos y claves de registro.

4º Se añade al registro e intenta suplantar un proceso propio del OS de Windows.

5º Manda comandos y recibe información sobre nuestra máquina.

Creo que queda a la vista que el ejecutable con el que estábamos tratando es un malware, vamos a ver si nuestro amigo Antonio puede explicarnos a qué se debe esto.



De momento esta clase acaba aquí. espero que les haya gustado, de momento pueden intentar seguir explorando las entrañas de este Troyano.

Saludos.

¡Happy Hacking!

Pueden seguirme en Twitter: @RoaddHDC

Si quieren hablar con otros seguidores por Telegram: @hackingde0

Para obtener actualizaciones rápidas por el canal de Telegram: @hackingd0

La página de Facebook: www.facebook.com/roaddhdc

Contactarse por cualquier duda a: r0add@hotmail.com

Si creen que el hacking también es arte: https://www.instagram.com/secureart/

Para donaciones, pueden hacerlo en bitcoin en la dirección siguiente:

1HqpPJbbWJ9H2hAZTmpXnVuoLKkP7RFSvw

También recomiendo que se unan al foro: underc0de.org/foro

Rolo.